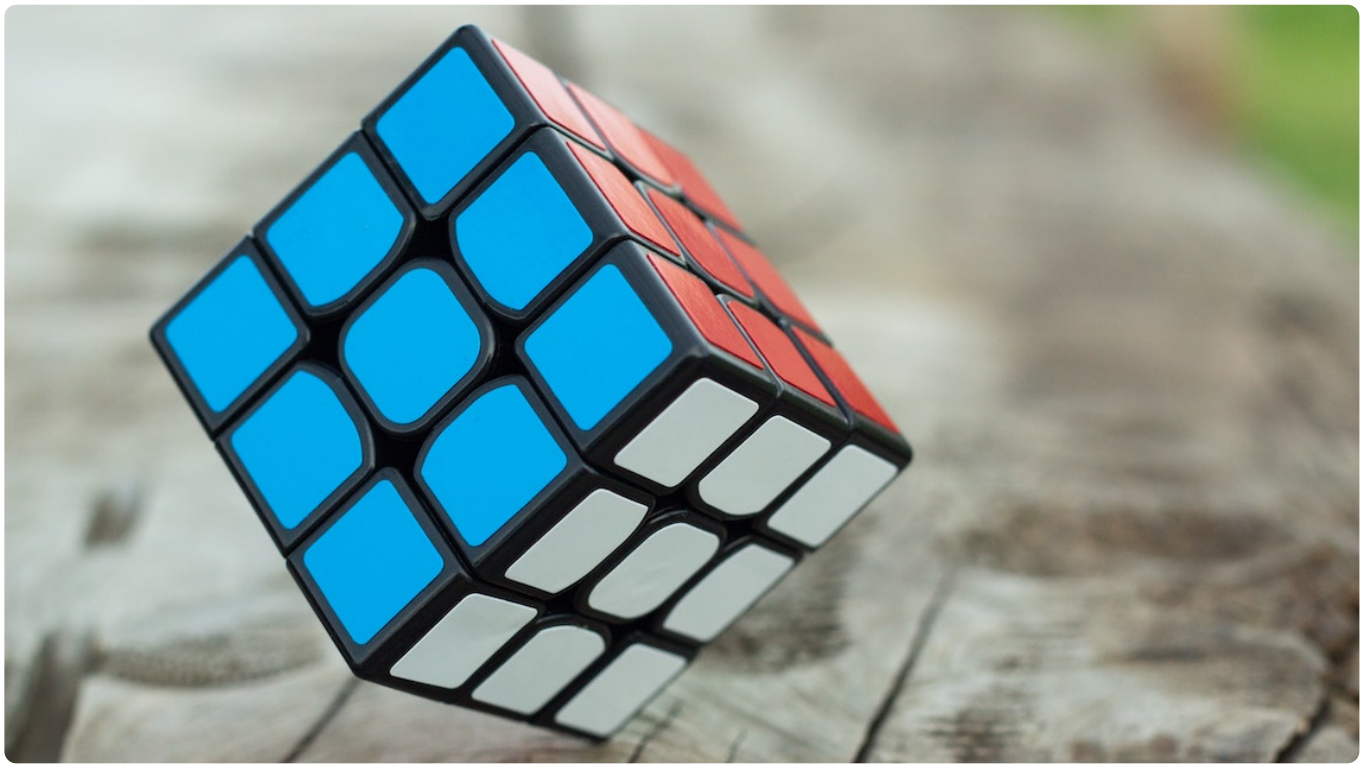
下载APP

答疑课堂01：面试计划篇热点问题解答

2021-04-24 四火

技术面试官识人手册 进入课程 >



讲述：四火
时长 13:37 大小 12.48M

你好，我是四火。

这一讲是关于常见问题的答疑课，对专栏第一大板块的“面试准备 / 计划”的篇章，还有一些很有意思的问题，在专栏正文中并没有得到足够透彻的讨论，那么我们就在这里把它们讲透。

问题 1：候选人经验比我丰富，我自认技术上也不如他，因而缺乏驾驭面试的自信怎么办？

这是一个非常常见的面试官心理建设的问题。尤其对刚做面试工作没多久的面试官，这个问题尤为常见。

首先需要明确的一点是，闻道有先后，术业有专攻。没有任何一条面试的原则说，你非要比候选人在某个特定的方面更加出色，才可以面试他。就是在一起工作的团队，其中的不同成员在特定业务和技术上，能力层次还有高有低呢，不过在做项目和沟通交流的时候，大家都是平等的。

其次，面试的过程，从你的角度看，就是使用你的方法考察候选人的过程，也是候选人考察目标团队的过程。

既然如此，把注意力放到你的面试具体实践上，无论是一起讨论一个问题，还是一起做一个小项目。在面试中，不妨把你们之间的关系，看做工作中的同事，一起解决一个业务或技术的难题。

再次，每个面试官都是这样过来的。据我所知，即便经过了相当长的时间，有些面试官还依然觉得战战兢兢，如履薄冰。

要知道有些情况下这是好事，缺乏自信可以让你远离自负，更加认真地对待面试这件事情，从做计划，到面试实施，再到之后的决策。

最后，最开始遇到这种情形时，有些面试官可能会紧张，这是完全正常的。事实上，无论紧张的原因是什么，不要去刻意地尝试克服紧张情绪，而是坦然地接受它。

再说了，适度的紧张还能够帮助提高专注力，我们只需要把注意力尽量转移到面试的流程和步骤上。相信经过多次的实践以后，我们就会慢慢淡定下来了。

问题 2：怎样看待脑筋急转弯类问题？

三个字，不推荐。

脑筋急转弯类问题，英文里比较接近的词是 Brain Teaser，指的是一类题面比较简单，但是解答需要跳出常规思维，给出“出乎意料”答案的一类问题。

原本这类问题就是问着玩儿的，但是后来它被用到了一些面试中去，甚至有不少是大公司，网上有很多关于大厂拿脑筋急转弯面试候选人的故事。

最开始，脑筋急转弯就是一些“娱乐性”很强的问题，但后来慢慢演变出更广泛的形式，看似更考验候选人的思维能力，比方说：

> 填满一辆校车需要多少个高尔夫球？

不知道你听过这个问题没有，这是网上流传得颇为广泛的一道题，据说是 Google 拿来面试候选人的。

我想说的是，我们只讨论针对软件工程师的面试的话，其实 Google 的技术面试的流程和其它几家互联网大厂都是类似的，如今说他们问脑筋急转弯问题是无稽之谈。而在一些互联网大厂的面试官培训中，都会强调一句话——不要问脑筋急转弯问题。

那我们进一步解释一下这是为什么。还记得我们在第 1 讲中就介绍了，面试的目的就是为了**寻找“合适”的技术人才**吗？

正因为这样，我们才那么希望面试尽可能还原实际工作的场景，这才有了希望在面试中“讨论一个问题”和“做一个迷你项目”这样的目标，这样收集到的数据，才能比较有效地判断候选人是不是“匹配”的。

而脑筋急转弯呢？即便它能考察候选人的思路是不是开阔，反应是不是迅速，但因为**它远离了我们要解决的实际问题，远离了我们的工作场景，既没有业务覆盖，也没有技术覆盖，它在技术面试中的价值实际就很低了。**

事实上，这些年来，有许多研究者仔细地分析了脑筋急转弯在面试中的应用，他们发现候选人对于这类问题的回答，和他们在实际工作中的表现缺乏相关性。

比方说这篇论文，它提到在面试中使用脑筋急转弯问题“缺乏有效的证据，且会令候选人感到不安”：

> lacks evidence for validity and is unsettling to job applicants

问题 3：怎样看待系统估算类问题？

系统估算类问题，是一种特定的考察候选人对软件系统理解的问题，它要求候选人估算一下在指定场景中，系统的规模的大小。比方说：

> 已知 Twitter 2020 年大约有 2000 亿的推文（tweets），如果你来设计 Twitter 系统，请问发推服务的吞吐量需要多少，网络带宽要占用多大，要存储它们需要多少磁盘容量？

这里说的是 Twitter 系统，但是类似的提问方式，套用到面试中讨论的软件系统，也是一样的。这个问题如何分析和求解呢？我可以给你一个简单的参考思路：

> 吞吐量，一般可以估算 TPS（Transaction Per Second），2000 亿 / 365 天 / 24 小时 / 60 分 / 60 秒，大概是 6000 左右，那么考虑 TPS 的波动性，我们简单预留 3 倍的空间，认为系统的 TPS 需要预留到 20000 左右，就大致可行了。
>
> 网络带宽，我们假设平均每条推文 200 个字符，众所周知每个字符占用 2 个字节（B，Byte），每个字节为 8 个比特（b，bit），我们在这里统一使用 Byte，而不是 bit（这里说明一下，你的网络运营商为了数据好看，可能会用比特每秒，而不是字节每秒宣传它的带宽）。这样每条推文就占用了 200 * 2 = 400 Byte，因此带宽就是 400 Byte * 20000 吞吐量 = 8 MB 的带宽。这里其实从 B 到 MB 应该使用 1/1024 的倍率，但是估算就使用了近似的 1/1000 的倍率。
>
> 存储，2000 亿推文 * 200 个字符 * 2 Byte = 80 TB 一年。

对于系统估算类问题，最重要的是过程要符合逻辑，而估算的最后结果并不是最重要的。过程正确的话，结果的数量级基本上是八九不离十的，那么设计出的系统往往就是靠谱的。

实践上，这一类问题我以前尝试用过，但是和其它类型的问题类型相比，它对候选人的要求其实比较高。我通常在面试高级工程师级别以下的职位时，就不考虑使用系统估算的问题了。

从职位来看，如果职位需要比较强的架构设计能力，或者团队是做软件中的基础设施部分的，那往往就需要候选人具备靠谱且快速的估算能力。

问题 4：对于小公司来说，怎样在人才招聘中和大厂竞争？

严格说，这是一个招聘的问题，而不是一个单纯面试的问题。但是，对于小公司来说，负责面试的同事，很可能承担了更多的职责，甚至有很重的“招人”的压力，这是很多大厂的面试官无法感同身受的。

因此，对他们来说，怎样去和大厂竞争，这就成为了一个实际的、需要在面试前搞清楚的问题，而这个问题有很强的普遍性，因此我想在这里专门谈一下。

有位同学 lihp 在专栏中留言谈到：

> 最近公司在招聘新人，秉承公司一贯用人理念，精英式团队，对人员的素质要求高，C/C++ 的技术基础扎实，还需要是一个培养潜力的人，尤其是在招聘有经验的工程师岗位（入职即可快速适应工作，参与产品和项目开发），招聘的进度不及预期，年初至今尚未找到合适的人才。

> 岗位要求技能和能力比较综合，不存在过度关注算法或某一项专项技术的问题，以能力面试为主，只要面试者表现强烈的学习意愿、良好的学习能力和沟通能力，技术基础与经验相匹配即可。

于是我就问道：

> 就你提到的“尚未找到合适的人才”我也好奇，因为从你简单的描述看似乎要求并不算特别高。那么，你觉得招不到人到底是什么原因呢？

接着他答复道：

> 公司岗位对合适的人，综合要求较高，学习能力强，思维逻辑清晰，专业技能扎实，也就是说我们觉得合适的人也符合多数大厂要求，候选人可选择性多。

> 1. 品牌知名度不及大厂。相比于知名的互联网公司，我司主要从事 toB 软件产品开发，在细分行业内还略有名气，但普通大众压根不知道，品牌商和大厂是没法比；
> 2. 薪资竞争力弱，大厂给更多，即使相同或者略少，大多数候选人也会选择大厂；
> 3. 筛选得到的候选人少，优秀的人才通过线下、内推等渠道选择工作，我司主要是通过简历、猎头等渠道，符合岗位需要的人数量较少。

对于中小公司，这就是一个现实的问题。

我记得一位负责招聘的朋友说，招人也有“二八现象”，就是市场上长期流通的简历中，有 80% 都是来自那 20% 的候选人，而很多优秀的候选人，在人才市场上出现的时间都很短。

如果要通过常规的招聘网站、猎头等渠道找到优秀的人，必然要面临巨大的竞争，而那些相对优秀的候选人，也就比较容易拿到 offer，甚至多家 offer，这时候企业面临人才的竞争就比较激烈了。

再从这位朋友的描述看，他对他们需要的人才的标准叙述很清晰，对优秀软件工程师的认识也符合主流的招聘标准，但由于知名度、薪资竞争力等方面不及大厂，那么自然，招聘就确实是一件较为困难的事情了。

分析完问题，再说说问题解决的思路。

首先是要清楚自身的优势。知名度和薪资竞争力不及大厂，这是事实，那么也要看到，大厂往往有着大公司病，小公司也可以有它自己的优势，因此**差异化**是关键。

这个差异化既包括在对候选人的要求上有所区别，也包括提供的福利待遇和工作挑战有自己的优势。比如工作时间和地点灵活，在某些技术细分上顶尖，职位重要性高，工作影响力大，以及能提供未来兑现的期权等等。

无论如何，不和大厂硬刚，寻找自身的优势，只要双方的需求能够得到大致匹配，那么候选人就能招得进来，也能留得住。

我举个自己主导面试的例子吧。大家都知道，无论是从国际上看，还是在北美，Amazon 的 AWS 在云计算领域都遥遥领先，而 Oracle 的 OCI（Oracle Cloud Infrastructure）进入市场晚得多，份额也显然不能同日而语。因此我经常被候选人问到一个问题，作为软件工程师，在 OCI 工作和 AWS 工作相比，有哪些优势？

我会谈到几个方面，其中一个是流程，我经常举一个真实的例子，一位朋友在 AWS 的 S3 组工作，他修改了一行普通的代码，需要经过好多个组的评审，花了整整三个月才让他的代码上线。而在 OCI 就不会有这样的情况发生，流程更为敏捷和快速。

另一方面，如果要讨论影响力（impact），很明显一个相对小一些的组织，软件工程师更需要承担更重要的职责，也往往能在组织中带来更大的影响力。

再者，相较于成熟的 AWS，在 OCI 要解决的问题，往往是模糊的，很多问题更“新”，也更考验软件工程师解决实际问题的能力，这一点很锻炼人。

其次，小公司尤其要扩展招人的渠道。以前我的老板说过一句话，叫做“时刻在招人”。线下的招人渠道，通常竞争更少，比如线下的技术活动，或者是因为开源项目而认识的朋友等等。

我有位朋友正在创业，他的团队能力很强，有意思的是这些人几乎没有一个是通过常规的猎头渠道，或者招聘网站渠道招进来的。

那好，今天就谈这几个问题吧，希望你有所收获。如果你有其它问题，也欢迎你在留言区继续跟我交流探讨，我也会和你聊聊我的看法。

好，我是四火，我们下一讲见。

提建议

© 版权归极客邦科技所有，未经许可不得传播售卖。 页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

上一篇 04 | 问题设计（下）：五个技巧助攻技术问题设计

下一篇 05 | 流程把控：控好流程，让面试进程高效有温度

精选留言

写留言

> 由作者筛选后的优质留言将会公开显示，欢迎踊跃留言。